Stephen Frueh

CS 499

E-Portfolio

Algorithms and Data Structures

For my selection to display my ability in algorithms and data structures, I chose to use my final project from my CS 260 class, which involved implementing various data structures, as well as algorithms to operate on them. For this assignment, I created four different data structures: a binary search tree, a hash table, a linked list, and a vector class. For each of these data structures, I implemented all the algorithms needed to effectively use each structure, such as inserting, deleting, and searching for various entries. Also in each file for each individual data structure is a test application using a Bid class used to effectively showcase everything the data structure is capable of doing. To confirm each one works, but also as an insight into exactly how they work, each data structure has various functions to print out functions, including various sorting algorithms for each data structure.

I decided to include this piece into my portfolio because being able to understand and effectively use algorithms and data structures is crucial to nearly every aspect of software programming. While most data structures might seem similar and the differences might not seem huge, each data structure is an individual tool in a programmers toolbox, and knowing exactly when and how to use each one is the most important part of having this variety of data structures. Each one has it’s own situations where it’s the correct tool for the job, and knowing which one that is and why will go a long way when programming, especially if speed and efficiency of code is a concern.

I feel this piece is a great choice to meet my portfolio entry requirements for data structures and algorithms. With some code refactoring to clean up the code base, and some more thorough commenting, I feel this piece is certainly ready for entry into my portfolio. In a real world scenario, I might choose to remove the code that tests and shows how each data structure works. Since this is a standalone example of various data structures and algorithms, though, I chose to leave it in for demonstrative purposes as well as confirmation that the code functions appropriately. It seemed at first like a lot of extra code, especially that it’s essentially duplicated across each file, but I felt that the code for the data structures was clearly separated and clear to tell apart from the application used to test and demonstrate the code. Each file also serves as a small application that can thoroughly go through each data structures algorithms, so it felt more appropriate to leave it included.

While I understand how these data structures and algorithms work, and I implemented them myself in the code, when not using these data structures and algorithms regularly, it can be easy to forget some of the details and nuances of each individual data structure or algorithm. Going through this code to enhance it and prepare it for my portfolio, I was able to refresh my understanding of each data structure and it’s accompanying algorithms, and exactly how each of them works. Especially going through and providing more detailed documentation, I was able to refresh my memory as to exactly what each data structure and it’s algorithms do.